Weekly Report

Lu Junhua

2015 年 9 月 28 日

This week, I mainly focused on data processing of netease. All the files related to the buy_shizhuang.txt are included.

At the beginning, I use the very traditional way to process the data, it was quite slow. Then I adopted MySQL database. Simply coping with buy_shizhuang data is easy and fast, it will not take much time while selecting the data I want.

During processing, I learned several skills :

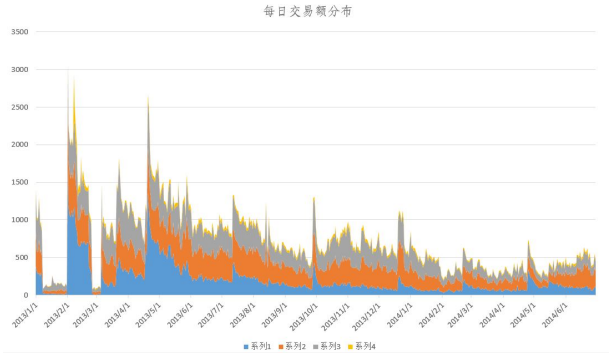
- If the number is too large and the difference between numbers are large, I may use $\log_{10}(N+1)$ to achieve better visual effects.
- If there are so large volumn of data to match, we need to use index to increase the efficiency of searching. If we want to use index again but for different dataset, we need to drop it first and then rebuild.
- If we want to filter out data under specific constrains, using boolean value is quite useful. Before I adopted cursor and it was quite slow.

```
75  select tmp.ymd,
76  sum(t1) h1,sum(t2) h2,sum(t3) h3,sum(t4) h4
77  from(
78  (select *,
79  hourofday>=0 and hourofday<=5  as t1,
80  hourofday>=6 and hourofday<=11  as t2,
81  hourofday>=12 and hourofday<=17  as t3,
82  hourofday>=18 and hourofday<=23  as t4
83  from netease.temp_proc) as tmp
84  )
85  group by tmp.ymd
```

- Another way to accelerate is using Hash Table. In java is hashtable or in python is dictionary. For example, we have 20w player_ids and hope to find the records of them in 4million records. We hash the playerids, and then read the records file line by line and write it to a new file. An example using python:

```python
import sys

fin = open("C:\\Users\\LJH\\Desktop\\neteaseData\\allid.csv", "r")

fin.__next__()

dic = dict()
for line in fin:
    dic[line.strip()] = 0

fin.close()
fin = open("C:\\Users\\LJH\\Desktop\\neteaseData\\2014-01-01.txt", "r")
fout = open("C:\\Users\\LJH\\Desktop\\neteaseData\\filtered.csv", "w")

for line in fin:

    _date,_time,serverid,source,target = line.strip().split(" ")
    if source in dic or target in dic:
        fout.write(_date+" "+_time+"\t"+serverid+"\t"+source+"\t"+target+"\n")

fout.close()
```

Following are some results of the data after processing.



Sometime excel may be better than D3.js...

The four time period is 0-5,6-11,12-17,18-23. The four price interval is 0-500, 500-3000, 3000-4000, more than 4000.

At present I am still loading data and tomorrow use python again to load the chat log information. More analysis will be shown next week. It really takes me much much time and I will cope with them in my best effort and speed.